**B2B Transport Console Toolkit (Python)**

A single Python 3 console app that bundles **10 independent logistics utilities** for a transport company prototype—covering registration, vehicle capacity, delivery time, pricing, route ordering, goods classification, tracking lookup, delivery stats, driver scheduling, and basic account management.

**What this code does**

* Demonstrates **practical decomposition** of a transport workflow into 10 small tools.
* Uses **introductory Python concepts** only (I/O, conditionals, loops, lists/tuples, simple functions).
* Focuses on **robust validation** and **readable structure** for easy testing and extension.

**Features (10 Tools)**

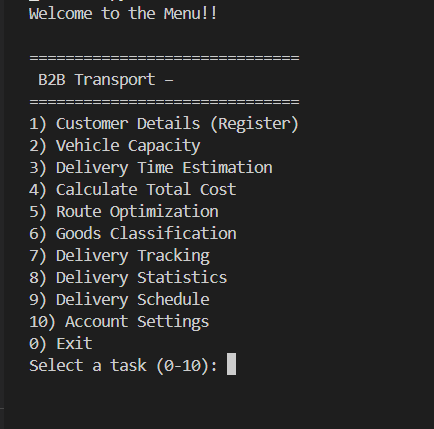
1. **Customer Registration** – collect details, validate (age > 21, 10-digit mobile), add to in-memory store.
2. **Vehicle Capacity** – L×W×H (m³) × 100 kg with a 5000 kg cap.
3. **Delivery Time** – distance/speed with +8h rest for long hauls (> 15h).
4. **Delivery Cost** – rounds weight to nearest positive int; $0.10 per kg·km; min $30; 5% off if distance > 100 km.
5. **Route Ordering** – sort 3 destinations by distance **descending** (stable tie-break by input order).
6. **Goods Classification** – converts pounds → kg; categorizes into 4 weight bands.
7. **Tracking Lookup** – validates a simple tracking format and returns status from sample data.
8. **Delivery Stats** – average, fastest, slowest across 3 trips (with destinations).
9. **Driver Scheduling** – evenly assigns N deliveries across 3 drivers (last driver may get remainder).
10. **Account Settings** – email+password login; update name/email/password; set newsletter/promos using **parallel lists**.

**Tech & Constraints**

* **Python 3**, standard library only.
* Data stored in memory using **parallel lists** (for simplicity and transparency).
* Minimal helpers for input parsing and validation (e.g., int/float readers, email/mobile checks).

**Getting Started :  
run : python Transitops\_Toolkit.py**

Use the menu (0–10) to run each tool. Tip: run **(1) Registration** first to create an account, then test **(10) Account Settings**.

****

### Data Model (in memory)

* user\_names, user\_emails, user\_passwords, user\_prefs (parallel indices).
* Sample tracking lists for quick demos: tracking\_numbers, tracking\_statuses.

### Validation Highlights

* Email (basic check), 10-digit mobile, age calculation from DOB (simple logic—no datetime import).
* Weight rounding to **nearest positive** integer before cost math.
* Speed must be > 0; capacity capped to **5000 kg**; route ordering stable for equal distances.
* Driver license check: alphanumeric length (configurable).

### Limitations (by design)

* No persistence (no files/DB); data resets each run.
* Simplified validators (e.g., emails/licenses are basic).
* Tracking data is a small sample list for demonstration.

### Roadmap / Ideas

* Persist users and tracking to JSON; add simple auth hashing.
* Replace parallel lists with dict/dataclass or a lightweight repository.
* Add unit tests and CI; parameterize rules (caps, rates, discounts).
* Expand tracking with timestamps and status transitions.

B2B Transport Console Toolkit (Python) — 10 logistics utilities (registration, capacity, delivery time, pricing, routes, classification, tracking, stats, scheduling, account settings). Pure Python 3, standard library only. Clean structure, strong input validation, and easy to extend.

**Screenshot:**



